2015

The RIT SDN Testbed and GENI

Bruce Hartpence

The RIT SDN Testbed and GENI

Bruce Hartpence
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York 14623
Email: bruce.hartpence@rit.edu

***Abstract*—Software Defined Networking (SDN) is an innovative approach to network construction, configuration and monitoring. While it is relatively new (most of the ground work was completed in Stanford circa 2008) it is transforming our thinking regarding communication architectures. SDN leverages several other technologies such as virtualization and integrates virtual and traditional networks. It is critical that researchers in the communication and networking spaces understand, utilize and finally experiment with SDN based topologies, whether they be local testbeds or remote general computing facilities. RIT does not possess many non-production switches or routers capable of communicating via OpenFlow, the devices must be virtualized. As with many compute resources, the RIT systems are typically geared towards providing processing power, not experimentation with connection technologies. For this reason, and because it provides an excellent educational opportunity, a testbed would be built. This paper includes a general description of the process and milestones but that complete build is documented in . A smaller version of the testbed would be attempted by students later on.**

This project experimented with SDN topologies through the construction of a local testbed. The same experiments were run on the Global Environment for Network Innovation (GENI) in order to determine the practicality of the two approaches for SDN experimentation and education. Students would then be given an opportunity to try the same builds. While the experiments can be run successfully in both venues, there are definite pros and cons regarding each approach. In addition, student success is highly dependent on their previous hands on experience. This paper documents this project and it's findings.
KEY WORDS: SDN, IoT, virtualization

I. Introduction

Software Defined Networks (SDN) and Network Functions Virtualization (NFV) represent ground breaking changes to the way we build, monitor and troubleshoot networks. The major difference between SDN and current networking practice is that SDN exposes a flow table in each network device and separates the control and management planes of devices in the flow path [4]. The flow table is manipulated through a communication protocol called OpenFlow [6] that runs between the network device and a controller. Devices that support OpenFlow are called OpenFlow switches. SDN networks are also tightly coupled with virtualization which further extends its flexibility and adaptability. Table modifications range from simple layer 2 forwarding to complex tunneling and MPLS.

II. Motivation

What is interesting about this area and what drives the motivation of this project, is that the changes embraced by SDN impact an industry that has become stagnant. Network design, management and monitoring are very manual processes. Vision into the network and operations is very difficult because devices (routers, switches, etc.) while configured within the scope of a larger architecture, are handled individually. Another pain point for administrators is orchestration. To say that automating a task or response across a complex topology is challenging would be a vast understatement. SDN embodies an approach that may radically change the body of technique used in all of these situations [5]. From an educational perspective, programs must keep up to date while maintaining the foundation of their disciplines. SDN and the related technologies build upon virtual networks and architectures such as those used in data centers and provider networks. Thus, developing SDN modules advances the program while reinforcing the underlying ideas. But SDN is not a "first step" technology as there are building blocks that go before. So, an important question is whether or not the students attempting to implement these new ideas truly understand what they are doing. What is the level of student success that can be expected?

The problem is that in order to experiment with SDN or SDN enabled technologies/ideas the researcher must have either SDN enabled devices (such as those manufactured by Brocade, HP, Nicera or BigSwitch) or have access to SDN virtual machines or VMs. For example we might use Linux VMs and install controller/switch packages such as POX and OVS. When working with VMs, compute resources can be obtained from either a local or cloud facility. Cloud resources include Amazon Web Services or even private clouds. Locally we might use RIT clusters or build a smaller setting such as one based on VMWare ESXi. However, general compute resources is that they may not be appropriate for the task at hand. For example, clusters may be well suited for providing significant processing capability but they are not very efficient when the need is for a large number of VMs. Such is the case for the RIT compute resources. GENI (Global Environment for Network Innovation) is an NSF sponsored architecture built for the purpose of experimentation with SDN type topologies. However, the resources are time restricted and storage is extremely limited. An exploration of the architecture itself must be completed in order to ensure it's appropriateness [7]. One final possibility is to work with a tool called Mininet [3]. Mininet allows the construction of

SDN topologies within a single virtual machine. While there are certainly limitations to this setup, Mininet can also be connected to physical topologies.

The contribution of this project will be to explore the issues associated with construction of an SDN infrastructure. An examination of remote facilities will also be completed and the two approaches will be compared. Specific project goals include:

- Develop a list of necessary hardware and software components.
- Deploy an ESXi hypervisor and interconnect it with the wired network.
- Deploy the SDN controller and the associated openvswitch.
- Complete SDN experiments on both the testbed and GENI
- Develop a comparison for researchers and educators.
- Perform some evaluation of the requirements for student success.

Over the course of the project the pros and cons quickly became evident. A local testbed presents opportunities for learning that far exceed those of remote and prepared facilities. On the other hand, to man-hours and expertise necessary to run a local testbed can be overwhelming. Student success rates vary quite a bit; for the testbed construction only 50% of the students attempting the build were successful. When using the GENI resources, more than 70% completed the experiments.

III. Classes

Over the course of this project there were two classes that we asked to use the testbeds, GENI or both. NSSA 602 Enterprise Computing is a graduate class delivered in an online format. In the course of their studies, GENI was an example of a large scale infrastructure. In order to give them a little experience with virtualized compute resources they were required to complete two GENI tutorials as assignments.

The second class was an SDN seminar composed of graduate and undergraduate students. This class was taught in the networking labs at RIT and so the students had access to physical equipment. In this case the student were asked to both build a smaller version of the testbed described in the paper and complete the same two GENI assignments.

IV. The Testbed

There are three major uses for the testbed; establishing the requirements for a locally administered facility, comparison with non-local resources and evaluating the educational requirements embodied by a virtual, SDN based architecture.

A. Testbed - Phase 1

Phase 1 was the construction of the virtualized infrastructure. This amounted to installation of VMWare ESXi, the base virtual machines, software and some portion of the physical network in order to allow management nodes to communicate with the virtual machines. Management access also had to be configured. The base topology is shown in Figure 1. In this image, the two virtual switches, networks and machines can be seen. Connectivity from the lower virtual switch can only be established through another VM that bridges the two switches. ESXi is a bare metal hypervisor that is managed

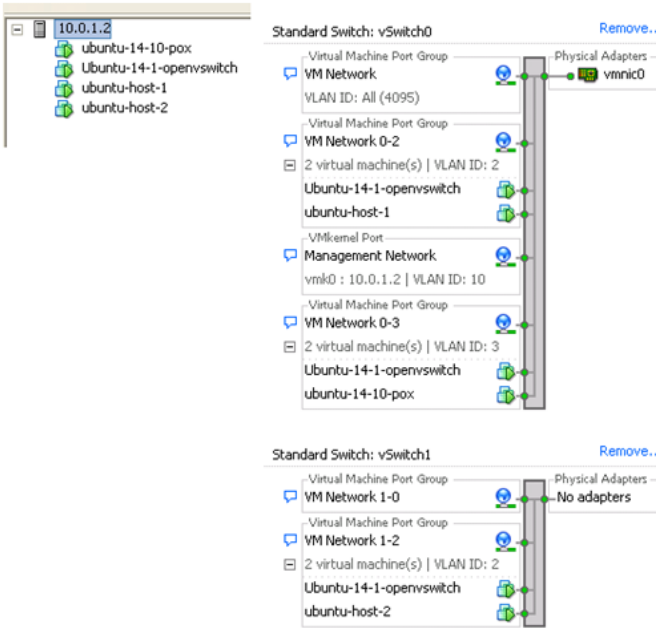

Fig. 1. ESXi Topology

remotely via VMWare vSphere. Ubuntu Linux was chosen as the base distribution for each VM due to its ease of use and the apt package manager. Four virtual machines were created; POX controller, OVS (openvswitch) and two network nodes. The POX controller was also chosen for its ease of use but also because the experiments could easy be duplicated on GENI. POX and OVS are packages that had to be installed and configured on top of Ubuntu. The virtual machines had to be connected from the virtual switch inside of ESXi to the hardware switch of the testbed. Lastly, the VMs must be allowed to connect through the SDN topology within ESXi to the outside world. This network configuration is non-trivial as it requires understanding of several advanced networking topics including VLANs, trunks, port forwarding and network address translation. At the conclusion of this phase, the first series of tasks for the project were completed. The final SDN topology can be seen in Figure 2.

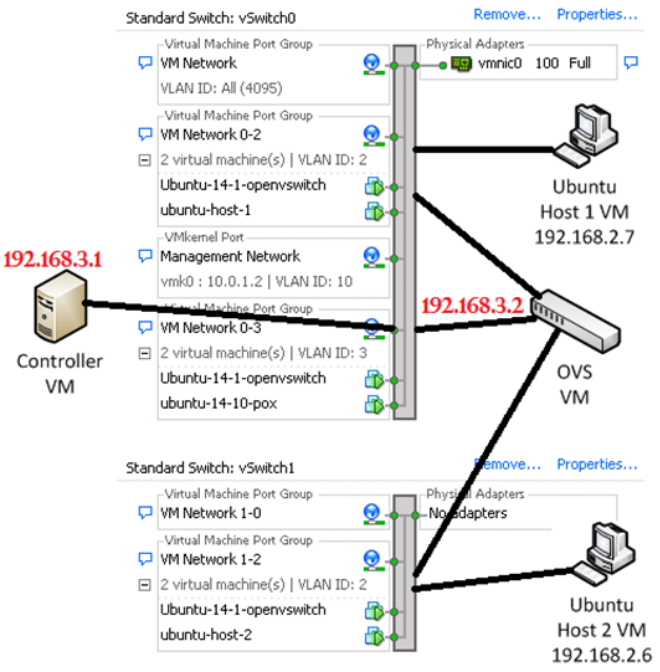

Fig. 2. Final SDN Topology

Once the topology was constructed, a collection of SDN experiments was selected for comparison against running them on GENI. These included layer 2 and layer 3 connectivity experiments.

B. Testbed - Phase 2

While the topology and tests were completed, there were still a couple of secondary activities that would increase the value of the project. One such task was to develop a methodology that might make the overall architecture less onerous to work with and therefore more practical for experimentation. To this end, the router and switch configurations were modified for eternal access and the security of the system was increased. Security improvements included installation and configuration of Secure Shell servers (sshd) on virtual machines and hardening of the router.

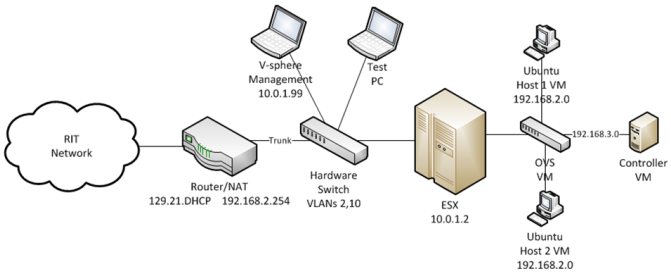

Fig. 3. Additional Physical Topology

Note that the networks, nodes and VLANs seen on the external network devices are connected to virtual versions within the ESXi chassis.

C. Testbed - Phase 3

As mentioned previously, Mininet could be used for SDN experimentation, especially when access to facilities is difficult. Mininet is running in a separate VM with a switch and series of hosts. The challenge is in integrating Mininet into the topology because two switches will seek to connect to the same controller. Mininet commands can be run interactively but it is more efficient to use the Python API and so a script was written to handle the tasks of the Mininet topology. The final topology depicting all of the nodes is shown in Figure 3. Students would not be working with Mininet for this project due to time constraints.

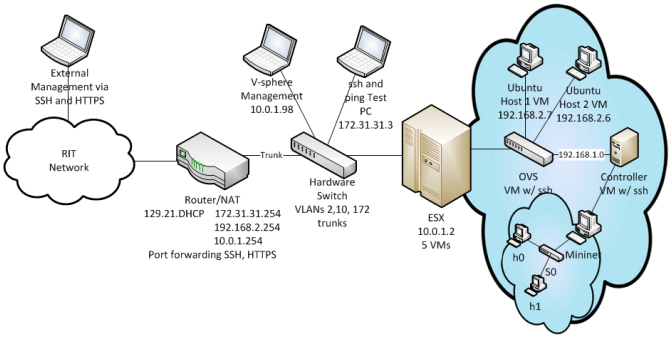

Fig. 4. Additional Physical Topology

As can be seen, the topology is growing in complexity. While not large (it consists of 6 physical boxes), the number of ideas and objects to manage can become daunting. Students that can master this are in a very good position to create further experiments or run larger, production systems. A sort list of the skills used include switch and router configuration, VLANs, addressing, DHCP server, network address translation, secure shell and operating system fundamentals. Another skill that is not as obvious but becomes increasingly important is the ability to think about topologies in a logical manner. Being able to visualize connectivity, network boundaries and flow paths is critical to fully understanding an architecture such as this.

V. GENI

The Global Environment for Network Innovation (GENI) [2] is a network testbed that is available to researchers and students alike. It is primarily used for network experimentation, especially SDN, though there is a wide variety of projects that use this resource. Fashioned after EMULAB and PLANETLAB, users are provided with a collection of time limited virtual machines. The virtual machines are typically Linux hosts and these can be connected in whatever fashion desired. The VMs can be modified and so users can also install software or run tests; and many of these activities can be scripted. In this project, the RIT use of GENI would be to compare some basic operations with the same operations run on the testbed. Further, students would attempt the same series of tests.

A. Experiments

GENI is well documented and has a series of tutorials posted on the website [1]. Two of these; Hello GENI and the L2 Forwarding were chosen for student experiments. Initial tests would simply determine if the experiments and their results could easily be run on both the GENI and testbed resources. Later students would be asked to complete the same tasks on GENI. A smaller group of students would build their own testbeds and attempt the same tutorials.

VI. Analysis

There are a couple of aspects of the architecture to be examined; the issues associated with building and running an SDN testbed, students building their own testbed and a comparison between locally administered and remote resources.

A. Testbed Construction

Simply completing the process of testbed construction is an outstanding learning experience for faculty and students alike. Students have the opportunity to install and manage their own virtualization infrastructure, connect this to a physical topology and configure layer 2 and layer 3 communications. However, it is time-consuming, requiring students to hold physical resources for an extended period of time which can make lab activities challenging. Simply obtaining the necessary equipment can also be a challenge. There is enough complexity to make troubleshooting difficult so that repairs to the SDN flows may have nothing to do with SDN itself.

There are number of technical challenges that crop up in a build of this sort including software/hardware failures, version mismatches, licensing, simple limitations as to what the gear can do and even power outages. Other issues that troubled this topology include poor documentation of source files, experiment design, and trying to truly understand the architecture and how it works. In some cases, work-arounds had to be deployed. For example, connecting to the ESXi box remotely through NAT was not supported and so a method of connecting to the VMs had to be configured. The GENI architecture converted to a new management system while the project was run and so this had to be learned. Even with all of this, if the architecture can be managed, students should be given this opportunity.

However, one of the original goals of the testbed itself was to help establish the requirements of a locally built architecture and determine the resources that would be required to run a class in which the students would have an in-depth, hands-on experience. Using the testbed we were able to determine what would be effective without being too time-consuming. The testbed trials and experiments were successful enough that a virtualization course will be deployed in the spring of 2016 based on this design. It should be noted that an important factor is that the system does not have to be high performance. The goal is to understand virtualization techniques and management and not to run servers. In our case the ESXi chassis was an HP computer with 8GB of RAM and a 250GB hard drive.

B. Testbed vs. GENI

One objective of this project was to determine the best architecture for use in experimentation and in class work. The obvious question was whether it was worth it to build the local testbeds. Two scenarios were to be considered; faculty and student experiences. Toward this end, several tests were run on both the local SDN testbed and on GENI. In the first case, the examination took into consideration the following;

- What equipment was required?
- How much time would it take for construction?
- What set of skills would be required before construction?
- What value was added over GENI?

Initially, the answer seemed very clear: the testbed would be best course forward. The learning and experimentation possibilities on a locally controlled resource that contained a number of open source tools seemed to far outweigh simply running a set of remote virtual machines. This opinion was not changed after faculty ran the GENI experiments and completed a series of performance tests on the testbed. However, this opinion did not survive contact with classes, or more accurately, all classes.

As mentioned earlier, there were two classes that used the GENI network. In the case of the on-line NSSA 602 Enterprise computing, the population consisted of graduate students. These students were required to complete two tutorials. This seems straight-forward however, there are a number of GENI concepts to learn and technologies with which students may not be familiar. For example, most students have used Secure Shell, but not all of them have had to manage public and private keys. In order to gage student success, three categories were examined; creating accounts/joining the project/completing system overview, tutorial 1 - a simple experiment and tutorial 2 which was more complex. Recall that the goal of this course was separate from the GENI SDN focus. In these three categories, 70% had complete success at the introductory work, with 30% having acceptable success. For tutorials 1 and 2 the numbers were 75% / 25% and 70% and 30% respectively. Similar results were seen in the SDN seminar class with all students managing to complete the tasks. One of the topologies used is shown in Figure 5.

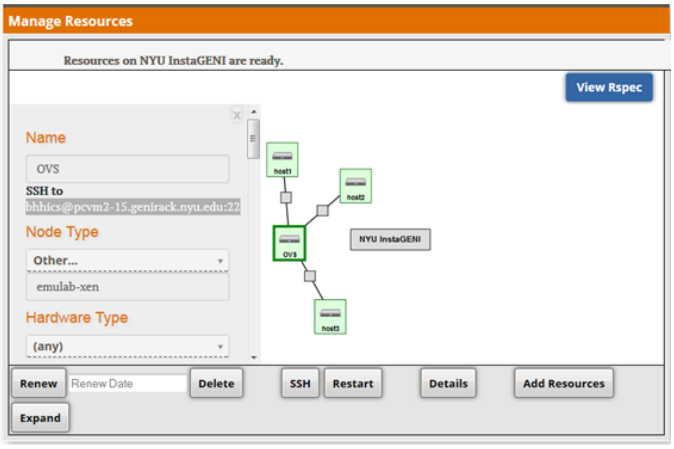

Fig. 5. GENI Tutorial Topology

This image is the topology drawing canvas used when interacting with a GENI aggregate. The students in the SDN seminar class were also tasked with completing their own smaller version of the SDN testbed. The Mininet and performance tests were eliminated from their builds. Once built, the testbed would then be used to run the same GENI tutorials but locally. Of the eight students in the seminar, only two managed to reach the end goal of attempting tutorial 2. Both of these were undergraduates, though there was one graduate student that did get the proper virtual machines deployed. The remaining students achieved some level of success with the level almost directly related to the amount of lab based classes they had previously taken. Taking all of this into account, we would estimate that the overall success of the student testbed project at about 50%.

Thus the prerequisite knowledge for the testbed becomes the limiting factor. For example, some of the students in the seminar could explain VLANs but had little experience with their configuration. They had even less experience with trunks. Class time had to be taken to explain these topics, putting the class a little behind. Some of these problems were expected but what compounded the problem was the differences in the student proficiency with the protocols and configurations. While the seminar was open to all students, core or required courses having similar content should definitely require a networking course prior to attempting a build such as this.

VII. Evaluation

Our experiences with GENI and the virtualized SDN testbed brought several interesting issues to the forefront. With regard to choosing between either a local testbed or GENI for use in classes or experimentation, there are advantages to both that should be weighed. The GENI project is highly beneficial in cases where the needed resources (machines, networking equipment) are not available or the local expertise is lacking. The same is true if the desire is for resources that require network nodes and connectivity. Normally connectivity is between virtual nodes resident on the same aggregate which has it's limitations. However, the GENI architecture team has recently improved the ability to connect between VMs on different aggregates. The means that a VM in NYU Poly can communicate with a VM in California. This opens up a host of wide area network studies. There are projects that are not necessarily best suited for GENI such as those requiring a great deal of storage or compute resources. These might be able to be run on a local testbed.

There can be minor performance differences because of the resources allocated for each VM and the hardware configuration of the hypervisor chassis. The term minor is used because there are only 5 VMs running on a chassis with 8GB of RAM. While this does have an impact, a larger number of VMs would have been more telling. So the preference comes down to available local resources vs. running experiments remotely. Attempting to manage and maintain remote resources can be a pain and there are resource limitations. For example, the type of VMs that can be used on GENI is limited. One notable advantage with the GENI architecture is that much bigger experiments can be run. In general, the same activities can be accomplished on either architecture.

As for using GENI in the classroom, the graduate students benefited not only from learning something about remote computing facilities but also had an opportunity to run their own experiments. The class used a discussion board that allowed student to help each other solve issues with connecting, keys and so forth. Since this was an on-line course, there was absolutely no downside to the experience. In addition, there was little difficulty in getting accounts created and, perhaps most important for some programs, it was completely free.

The case for classes the have access to lab facilities is less clear although with large topologies GENI might still be the best choice. Networking students often learn best when actually building topologies.Where any non-local computing facility falls short, and especially those not built by students, is that so much of the architecture and configuration is hidden from them. If the object of the exercise is to learn the underlying ideas and components, the testbed is a better way to go. Even if the students do not actually build it but can see greater detail regarding the configuration it would be an improvement over a remote facility. In these cases, GENI might be used to introduce or support class concepts.

The problems encountered during the seminar were more due to the nature of the course than anything else. Once the lab experiences are designed, an architecture like this is complex enough to require that student not only know the topics listed but have some level of proficiency. This would ensure a greater level of student success.

VIII. Conclusion

The paper describes an SDN testbed that was constructed at RIT. The testbed was used in an SDN seminar and is the model for a virtualization course to be offered in 2016. During the

same period students in this seminar and an on-line Enterprise Computing course used the resources of the GENI project. In comparing the two sets of experiences it was found that a good case can be made for each architecture depending on the goals of the class. Network experiments that were run on the testbed could just as easily be completed on GENI. However, the testbed exposes the students to greater configuration detail.

REFERENCES

[1] http://www.geni.net/.

[2] Chip Elliott. Geni-global environment for network innovations. In *LCN*, page 8, 2008.

[3] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.

[4] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[5] Rob Sherwood, Michael Chan, Adam Covington, Glen Gibb, Mario Flajslik, Nikhil Handigol, Te-Yuan Huang, Peyman Kazemian, Masayoshi Kobayashi, Jad Naous, et al. Carving research slices out of your production networks with openflow. *ACM SIGCOMM Computer Communication Review*, 40(1):129–130, 2010.

[6] OpenFlow Specification. 1.1. 0. *Specification for Master Minimum Equipment List (MMEL) Agenda Proposal & Coordination Process. Air Transport Association of America*, 1997.

[7] Katherine Yelick, Susan Coghlan, Brent Draney, Richard S Canon, et al. The magellan report on cloud computing for science. *US Department of Energy, Office of Science, Office of Advanced Scientific Computing Research (ASCR)*, 2011.